On the Effectiveness of Control-Flow Integrity in Practice

Lucas Becker, Jiska Classen, Matthias Hollick

August 13, 2024 - USENIX WOOT '24

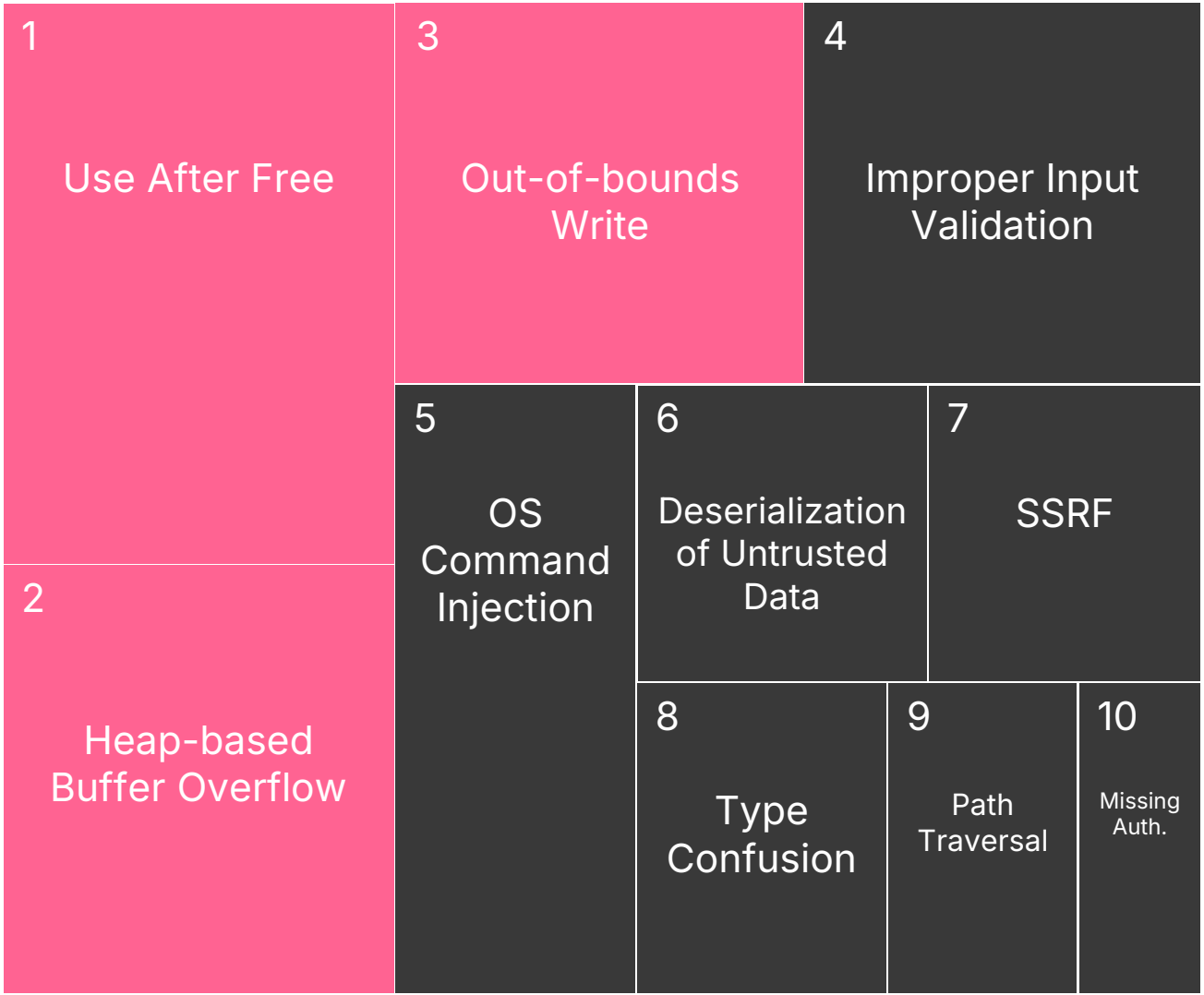
Why Control-Flow Integrity (CFI)?

“Memory Corruption Is a Solved Problem”

Why CFI?

MITRE
TOP 10 KEV 2023

46%



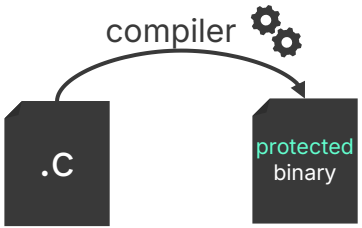
[1]

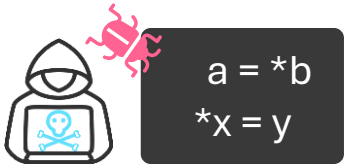
Why CFI?

“Memory Corruption Is a ~~Solved~~ Persistent Problem”

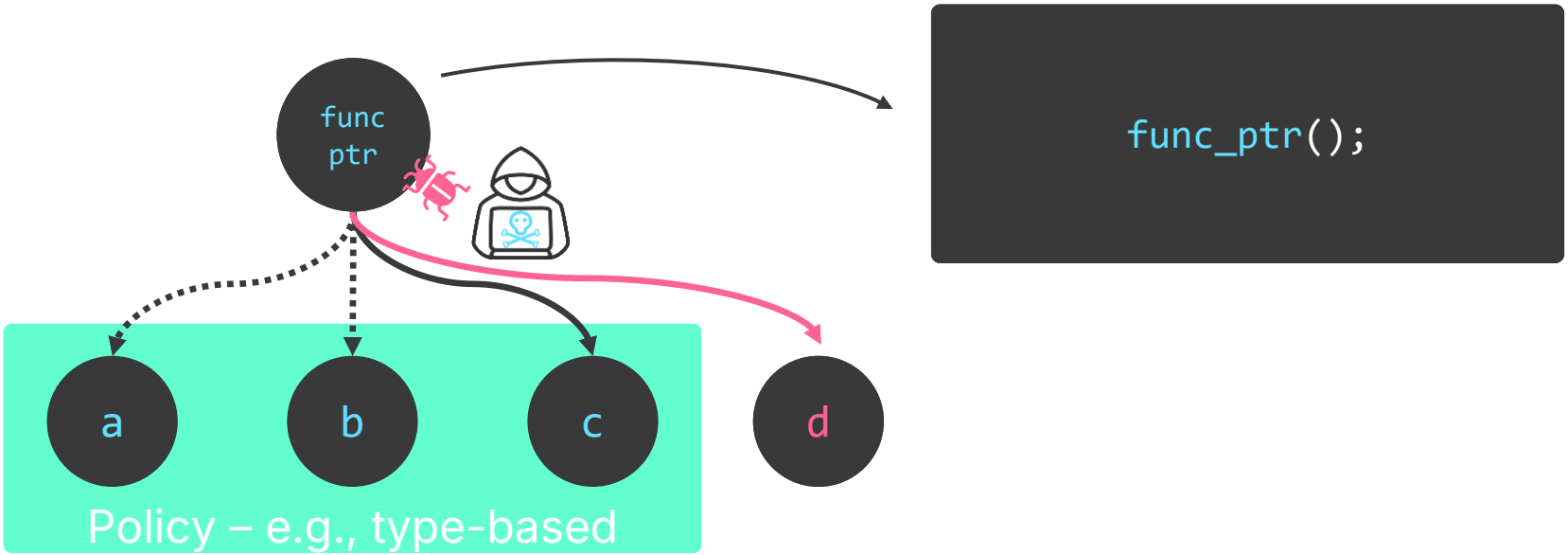
CFI as a Solution

CFI: Prevents exploitation of memory safety bugs
by inhibiting control-flow deviations

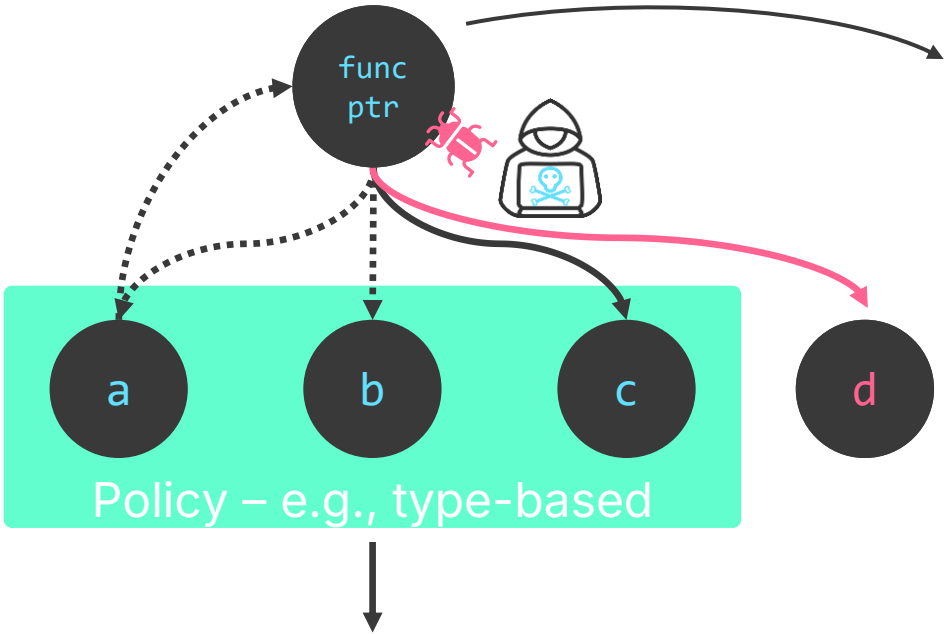




How CFI Works



How CFI Works



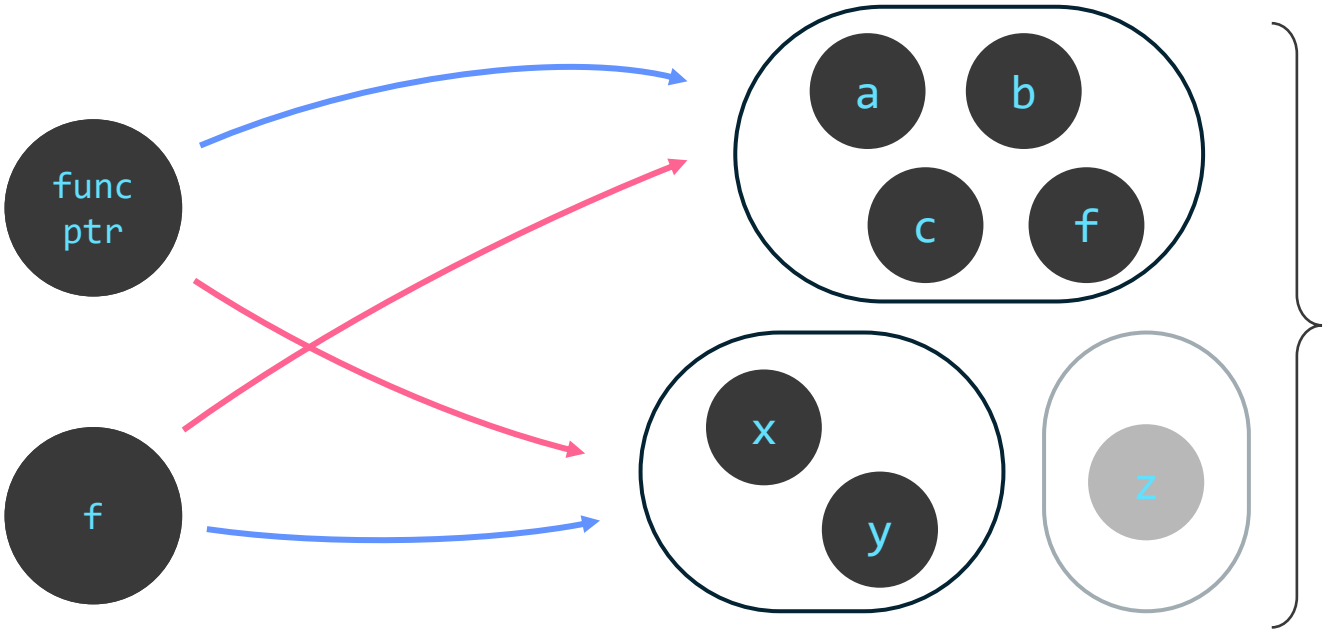

```
if (!InlinedFastCheck(
                    func_ptr))
   __cfi_slowpath(typeID,
                    func_ptr);
func_ptr();
```

```
__cfi_check(
        typeID,
        func_ptr);


      module
```

Control-flow Integrity + (Shadow Stack)

Equivalence Classes

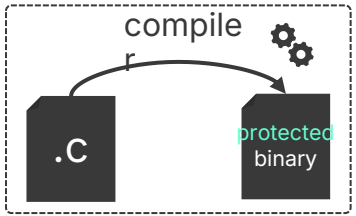


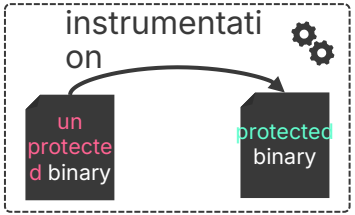
#EC fixed
↓
coarse grained

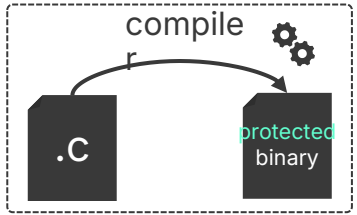
#EC program dependent
↓
fine grained

Academia vs. Industry

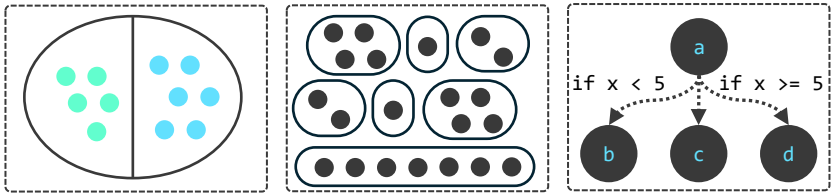
Approaches?

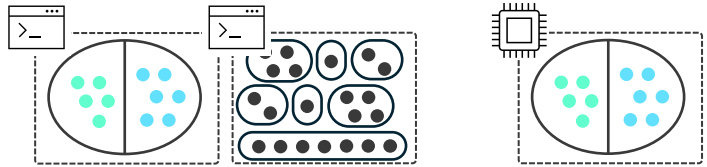






Granularity?





→ Academic schemes promising, but not (often) adopted

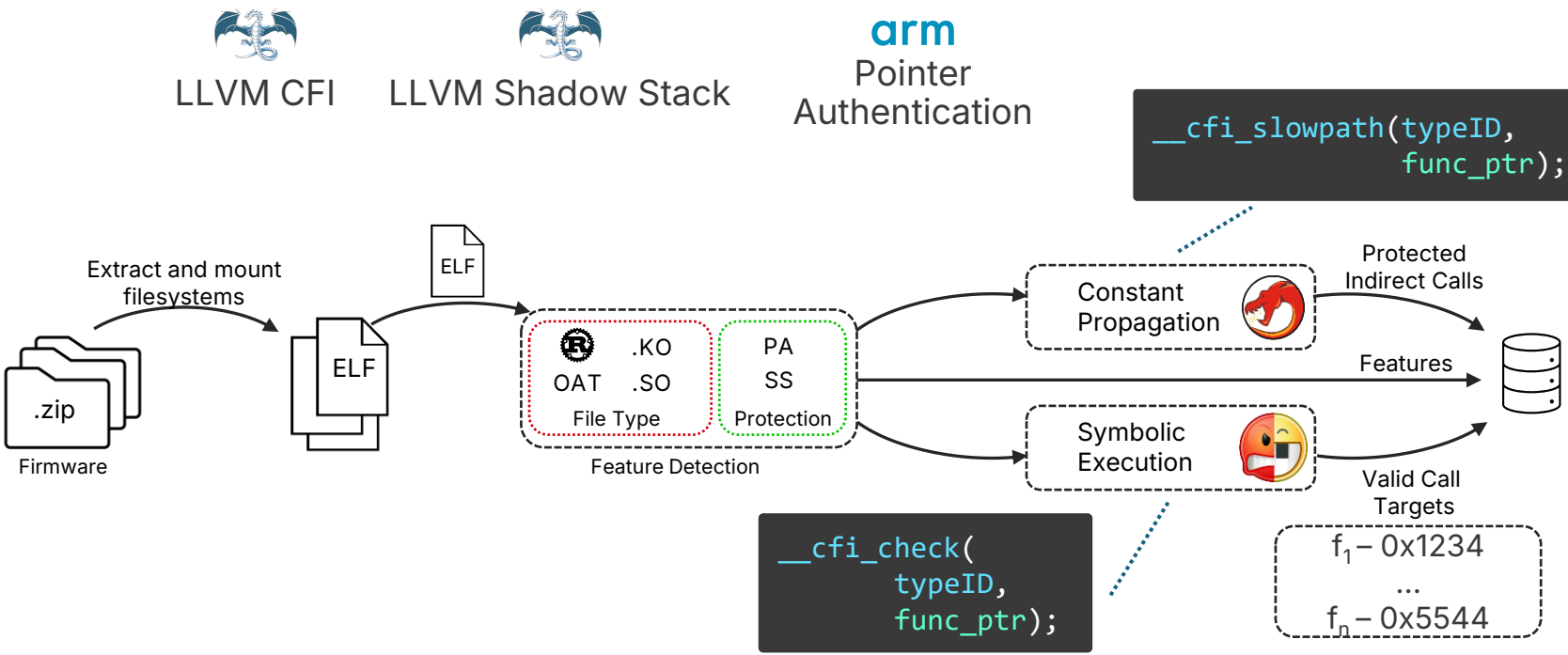
A Closer Look on CFI on Android

Supports a combination of different schemes

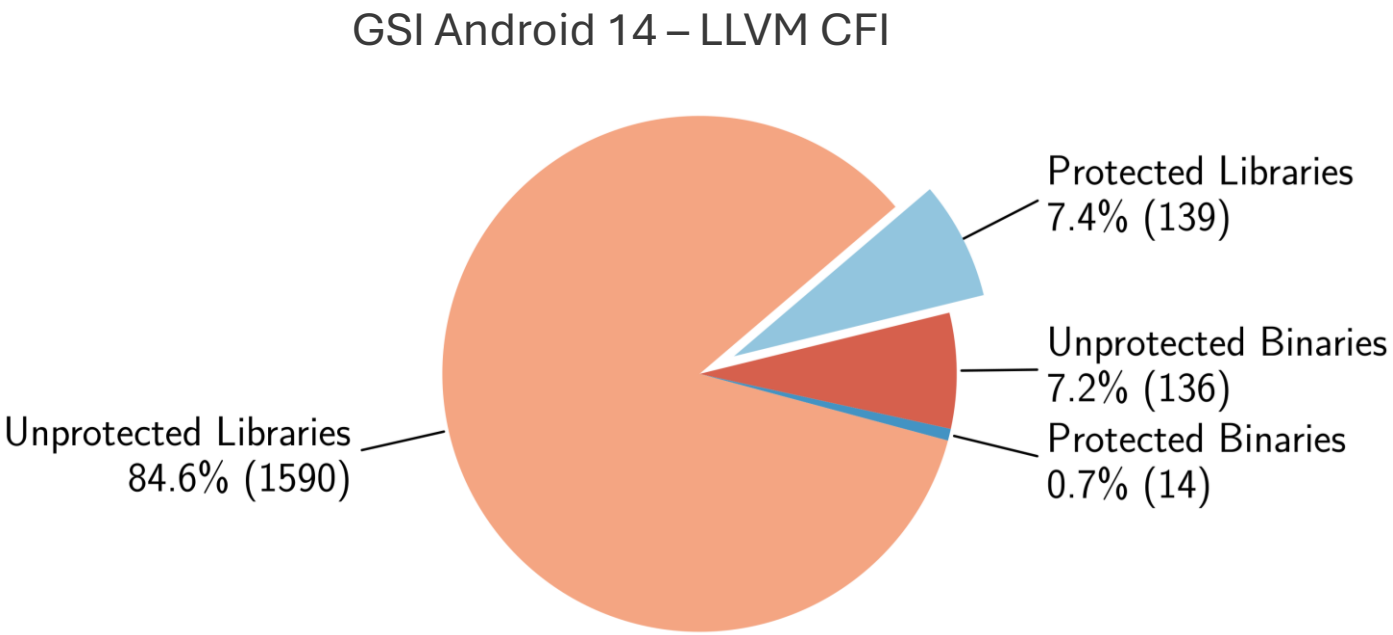
LLVM CFI

LLVM Shadow Stack

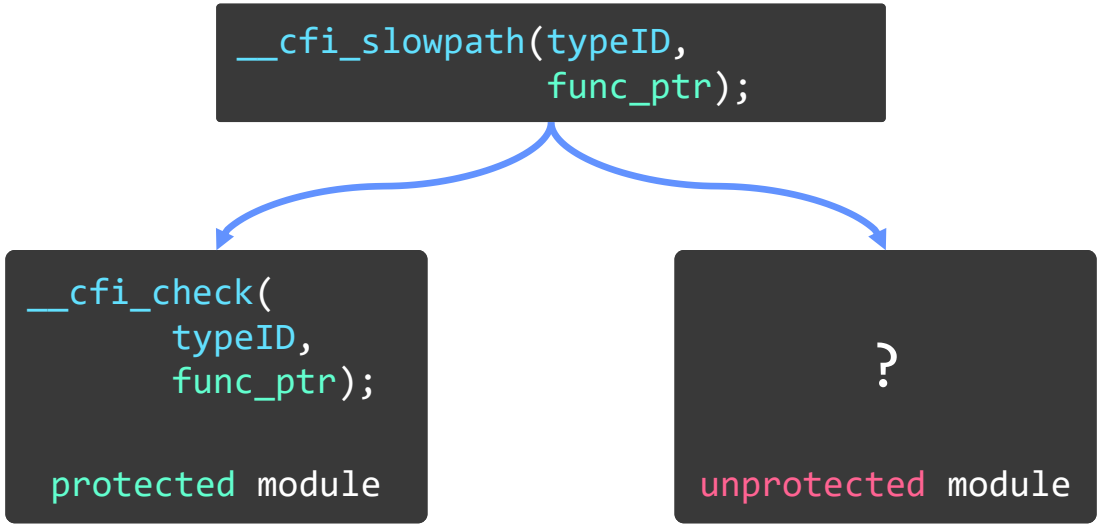
arm Pointer Authentication



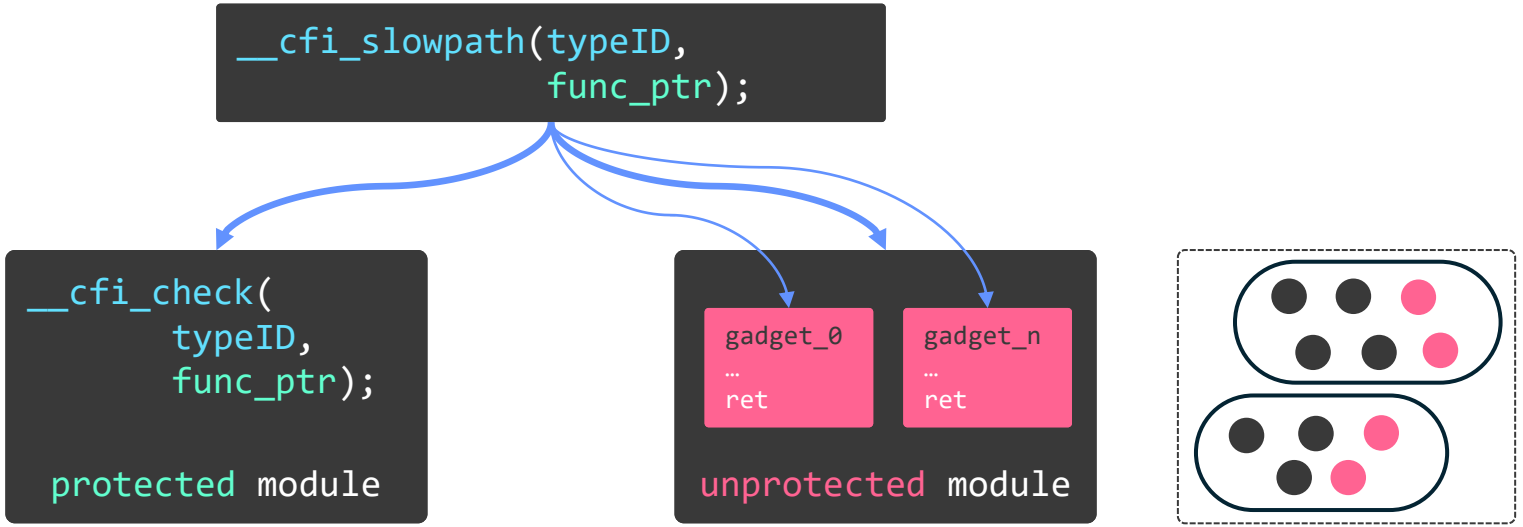
Android CFI Coverage



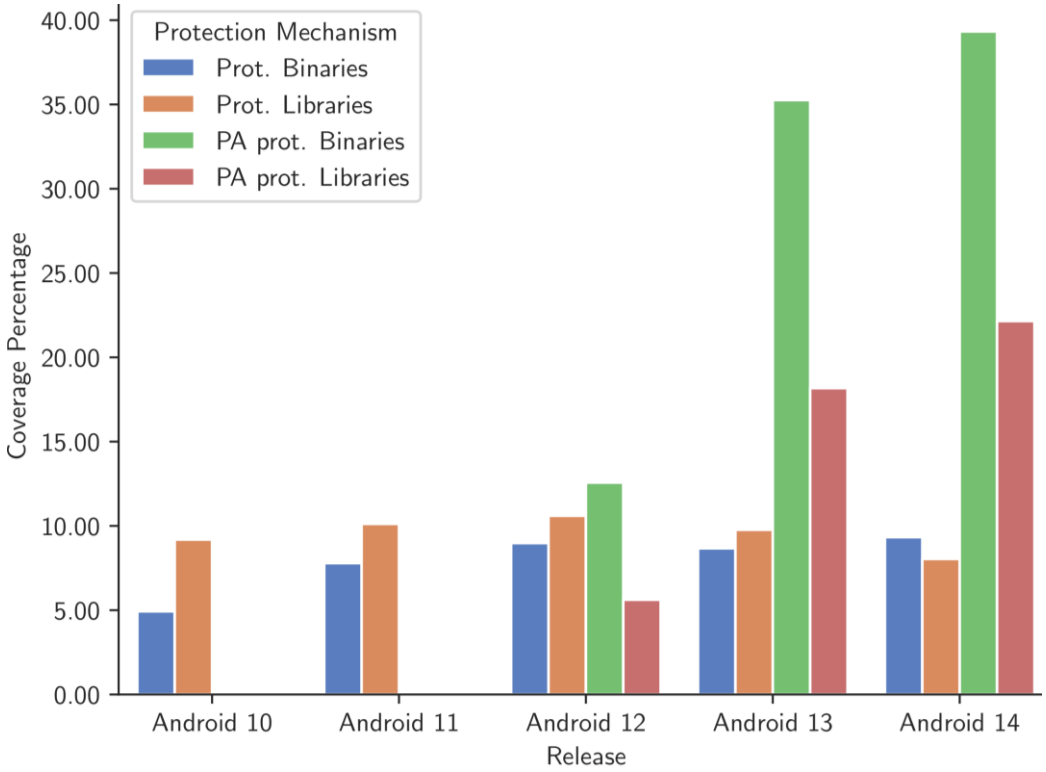
Unprotected Modules in LLVM CFI



Unprotected Modules in LLVM CFI



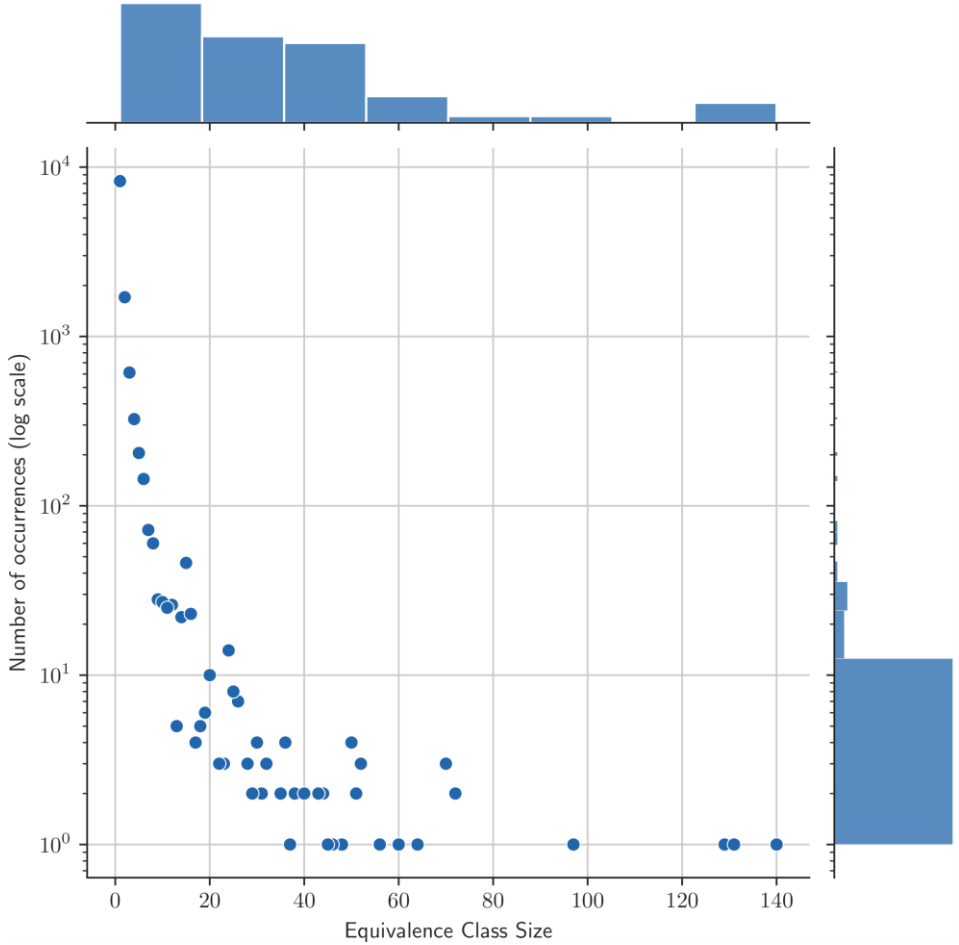
Android CFI Coverage – Over Time



Equivalence Class Size Distribution

GSI 14

Binaries & Shared Objects
(no dependencies)

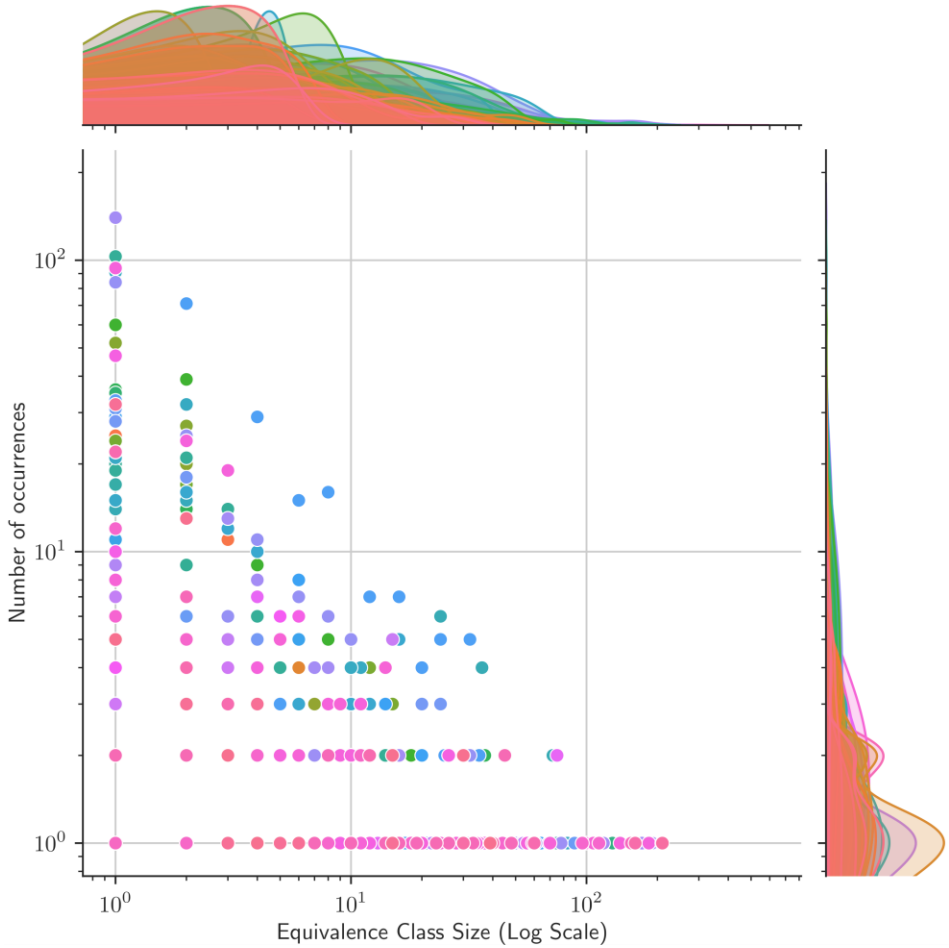


Equivalence Class Size Distribution

GSI 14

With dependencies & reachable

```
__cfi_slowpath(typeID,
               func_ptr);
```



On the Effectiveness of Control-Flow Integrity in Practice

Lucas Becker, Jiska Classen, Matthias Hollick

Takeaways

- Gap between schemes in practice and academia
- Android: Roll-out incomplete
- Equivalence class sizes look promising

In the paper

(eXtended) Control-flow Guard

Pointer Authentication

Branch Target Identification

CET Indirect Branch Tracking

CET Shadow Stack

The source code

https://github.com/seemoo-lab/woot24_cfi_coverage_tools/

References and Image Sources

[1]: MITRE, https://cwe.mitre.org/top25/archive/2023/2023_kev_insights.html#2023_kev_chart

Image Sources (by first appearances)

P. 3 – MITRE Logo: https://commons.wikimedia.org/w/index.php?curid=24829169
P. 5 – LLVM Logo: https://llvm.org/img/LLVMWyvernSmall.png
P. 5 – Visual Studio Logo: https://commons.wikimedia.org/wiki/File:Visual_Studio_Icon_2019.svg
P. 9 – Android Logo: https://en.wikipedia.org/wiki/File:Android_new_logo_2019.svg
P. 9 – arm Logo: https://www.arm.com/-/media/global/logos/Arm_logo_suite_2017 . The Arm word and logo are trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved.
P. 9 – Angr Logo: https://angr.io/img/angry_face.png
P. 9 – Ghidra Logo: https://commons.wikimedia.org/wiki/File:Ghidra_logo.svg
P. 9 – Rust Logo: https://github.com/rust-lang/rust-artwork/blob/master/logo/rust-logo-blk.svg
P. 15 – Windows Logo: https://commons.wikimedia.org/wiki/File:Windows_logo_-_2021.svg
P. 15 – Intel Logo: https://download.intel.com/newsroom/2021/corporate/2021-Intel-logos.zip Intel, the Intel logo and other Intel marks are trademarks of Intel Corporation or its subsidiaries.

All other icons are either self-made or part of the Microsoft PowerPoint Icon catalogue.